QA Tester Technical Assessment (Manual + Automation)

Harol Yovany Sastoque Gonzalez
Email: hard.sgo@hotmail.com

Section 1: Manual Testing

Test Case 1: Valid Form Submission

Field	Details
Test Case ID	TC-01
Summary	Verify successful form submission with valid data (positive flow).
Preconditions	User navigates to https://demoqa.com/automation-practice-form.
Steps	1. Enter First Name = "John". 2. Enter Last Name = "Doe". 3. Enter Email = "john.doe@example.com". 4. Select Gender = "Male". 5. Enter Mobile = "9876543210". 6. Select Date of Birth = "8 May 1985". 7. Select Hobbies = "Reading". 8. Enter Current Address = "123 Main St, NY". 9. Select State = "NCR" > City = "Delhi". 10. Click Submit.
Expected Results	Confirmation modal displays: "Thanks for submitting the form" with all entered data.

Test Case 2: Mandatory Field Validation

Field	Details
Test Case ID	TC-02
Summary	Validate UI feedback when mandatory fields are empty.
Preconditions	User is on the form page.
Steps	1. Leave Last Name blank. 2. Fill other mandatory fields: - First Name = "Jane". - Email = "jane@example.com". - Gender = "Female". - Mobile = "9123456780". 3. Click Submit.
Expected Results	Form submission is blocked. Last Name field shows red border indicating a required field error.

Test Case 3: Invalid Email and Mobile Validation

Field	Details
Test Case ID	TC-03
Summary	Validate error messages for invalid email and mobile formats.
Preconditions	User is on the form page.
Steps	1. Enter Email = "invalid-email". 2. Enter Mobile = "12345". 3. Fill other mandatory fields. 4. Click Submit.

Field	Details
Expected Results	Error messages display: - "Please enter a valid email" (under Email). - "Please enter a valid mobile number" (under Mobile).

Test Case 4: File Upload Functionality

Field	Details
Test Case ID	TC-04
Summary	Verify successful upload of a profile picture.
Preconditions	User is on the form page. A valid JPG/PNG file is available.
Steps	1. Click Select picture. 2. Upload profile.jpg.
Expected Results	The filename "profile.jpg" appears next to the upload button.

Test Case 5: State and City Dropdown Dependency

Field	Details
Test Case ID	TC-05
Summary	Validate dynamic population of Cities based on selected State.
Preconditions	User is on the form page.
Steps	1. Select State = "Uttar Pradesh". 2. Select City = "Agra".

Field	Details
Expected Results	Selected values: - State = "Uttar Pradesh". - City = "Agra".

Test Case 6: Multi-Subject Input Validation

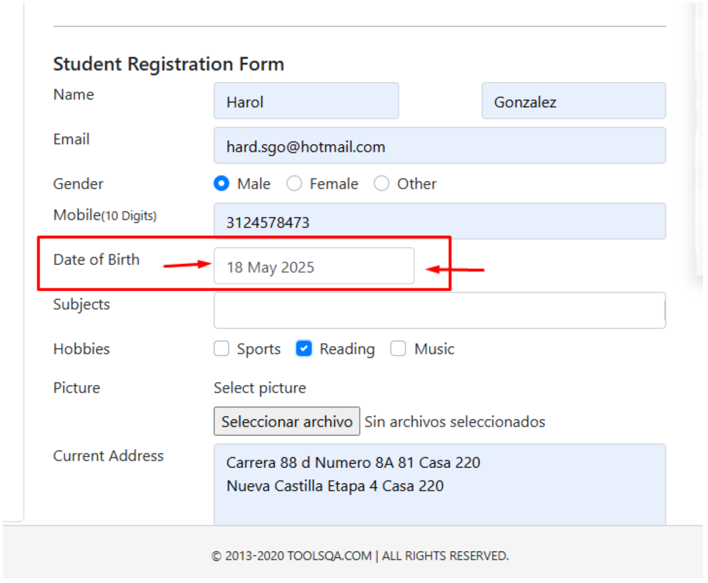
Field	Details
Test Case ID	TC-06
Summary	Verify the system allows adding multiple subjects dynamically.
Preconditions	User is on the form page.
Steps	1. In the Subjects field, type "Maths" and press Enter. 2. Type "Physics" and press Enter.
Expected Results	Both "Maths" and "Physics" appear as selected subjects in the field.

1.2 Bug Reporting

Bug 1: Email Field Accepts Empty Input on Form Submission

Field	Email
Title	Email field validation missing: Form submits successfully with empty email field.
Steps to Reproduce	1. Navigate to the form. 2. Leave the Email field empty. 3. Fill other mandatory fields (First Name, Last Name, Gender, Mobile). 4. Click Submit.
Expected Result	Error message: "Email is required".
Actual Result	Form submits successfully without any email address.
Severity	High (Mandatory field validation failure; leads to incomplete/invalid user data).

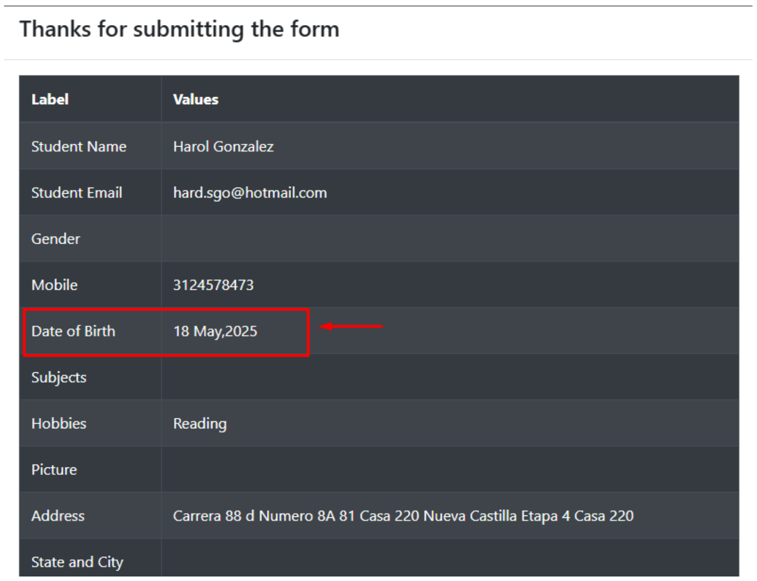
Attachments:



Student Registration Form

Name	Harol	Gonzalez
Email	hard.sgo@hotmail.com	
Gender	● Male ○ Female ○ Other	
Mobile(10 Digits)	3124578473	
Date of Birth	18 May 2025	
Subjects		
Hobbies	☐ Sports ☑ Reading ☐ Music	
Picture	Select picture Seleccionar archivo Sin archivos seleccionados	
Current Address	Carrera 88 d Numero 8A 81 Casa 220 Nueva Castilla Etapa 4 Casa 220	

© 2013-2020 TOOLSQA.COM | ALL RIGHTS RESERVED.



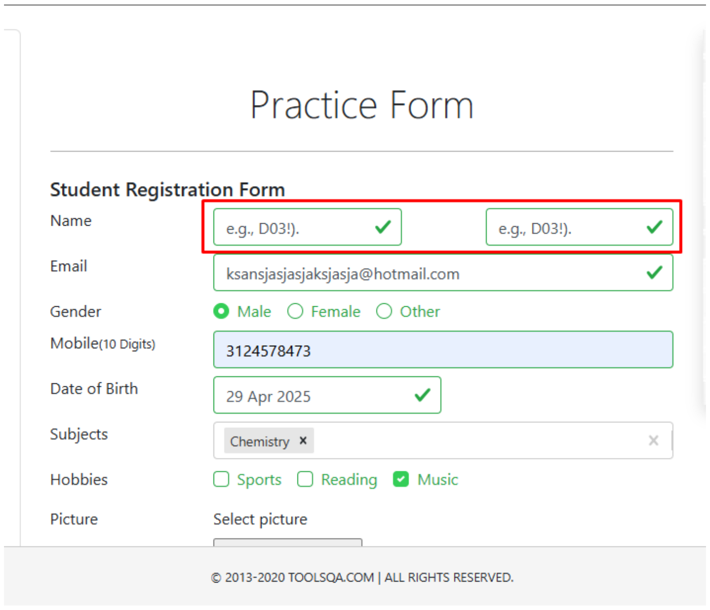
Thanks for submitting the form

Label	Values
Student Name	Harol Gonzalez
Student Email	hard.sgo@hotmail.com
Gender	
Mobile	3124578473
Date of Birth	18 May,2025
Subjects	
Hobbies	Reading
Picture	
Address	Carrera 88 d Numero 8A 81 Casa 220 Nueva Castilla Etapa 4 Casa 220
State and City	

Bug 2: First Name and Last Name Fields Accept Numeric/Special Characters

Field	Name
Title	Name fields allow non-alphabetic characters (numbers/symbols) without validation.
Steps to Reproduce	1. Navigate to the form. 2. Enter numeric/special characters in First Name (e.g., J0hn#) or Last Name (e.g., D03!). 3. Fill other mandatory fields. 4. Click Submit.
Expected Result	Error message: "Name fields must contain only letters".
Actual Result	Form accepts names with invalid characters.
Severity	Medium (Functional defect; violates input standards but does not break core functionality).

Attachments:

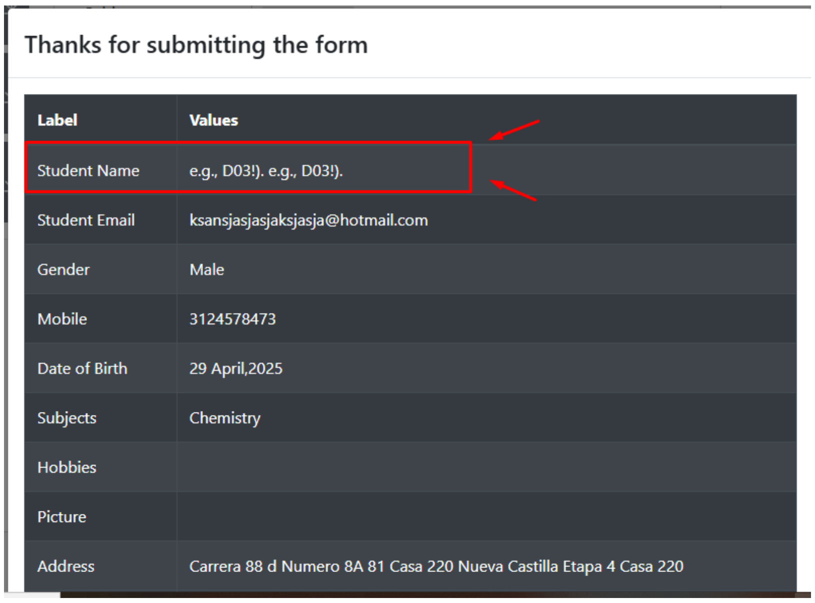


Practice Form

Student Registration Form

Name	e.g., D03!). ✓ e.g., D03!). ✓
Email	ksansjasjasjaksjasja@hotmail.com ✓
Gender	[✓ Male] Female Other
Mobile(10 Digits)	3124578473
Date of Birth	29 Apr 2025 ✓
Subjects	Chemistry ×
Hobbies	Sports Reading [✓ Music]
Picture	Select picture

© 2013-2020 TOOLSQA.COM | ALL RIGHTS RESERVED.



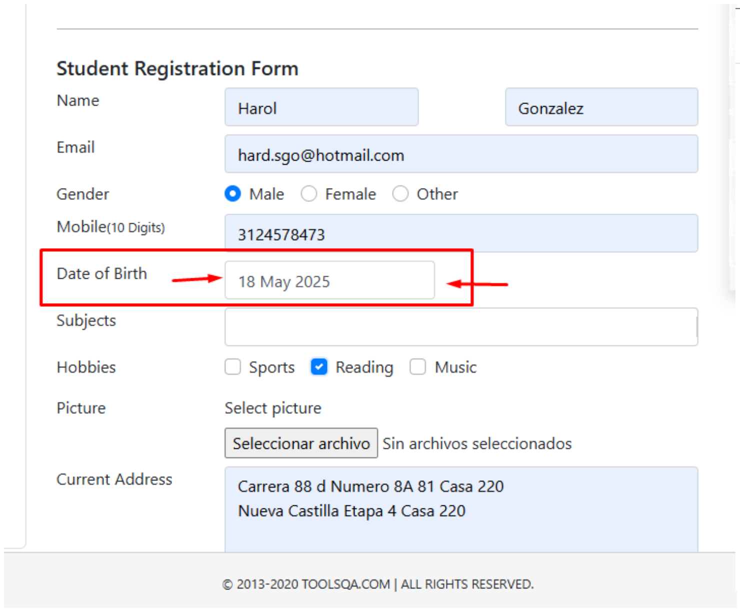
Thanks for submitting the form

Label	Values
Student Name	e.g., D03!). e.g., D03!).
Student Email	ksansjasjasjaksjasja@hotmail.com
Gender	Male
Mobile	3124578473
Date of Birth	29 April,2025
Subjects	Chemistry
Hobbies	
Picture	
Address	Carrera 88 d Numero 8A 81 Casa 220 Nueva Castilla Etapa 4 Casa 220

Bug 3: Date of Birth Field Permits Future or Underage Dates

Field	Date of Birth
Title	Date of Birth validation missing: Users can submit future dates or ages below minimum requirement.
Steps to Reproduce	1. Navigate to the form. 2. Select a future date (e.g., 18 May 2025) or a date indicating underage (e.g., 15 May 2020). 3. Fill mandatory fields. 4. Click Submit.
Expected Result	Error message: "Date of Birth must be a valid past date" or "Minimum age requirement not met".
Actual Result	Form accepts invalid dates without warnings.
Severity	High (Legal/regulatory risk).

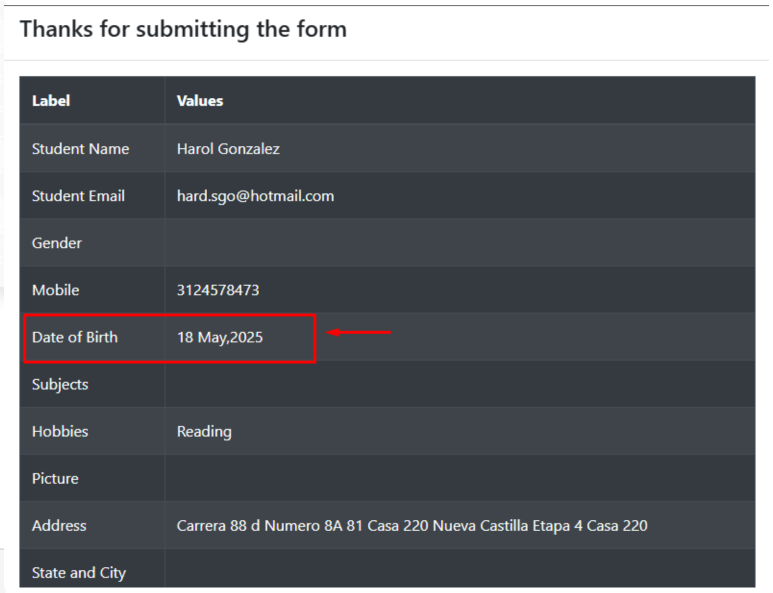
Attachments:



Student Registration Form

Name	Harol	Gonzalez
Email	hard.sgo@hotmail.com	
Gender	[✓ Male] Female Other	
Mobile(10 Digits)	3124578473	
Date of Birth	18 May 2025	
Subjects		
Hobbies	Sports [✓ Reading] Music	
Picture	Select picture Seleccionar archivo Sin archivos seleccionados	
Current Address	Carrera 88 d Numero 8A 81 Casa 220 Nueva Castilla Etapa 4 Casa 220	

© 2013-2020 TOOLSQA.COM | ALL RIGHTS RESERVED.



Thanks for submitting the form

Label	Values
Student Name	Harol Gonzalez
Student Email	hard.sgo@hotmail.com
Gender	
Mobile	3124578473
Date of Birth	18 May,2025
Subjects	
Hobbies	Reading
Picture	
Address	Carrera 88 d Numero 8A 81 Casa 220 Nueva Castilla Etapa 4 Casa 220
State and City	

1.3 Exploratory Testing Summary

The form has serious problems checking user inputs, which could lead to incorrect or unreliable data. For example, it allows invalid email addresses (like "user. Example" instead of "user@example.com"), numbers/symbols in the First Name and Last Name fields (e.g., "John1343"), and future dates (e.g., 2025) or dates for underage users in Date of Birth. Worse, users can submit the form even if required fields like Email or Names are left empty. This means the form does not follow basic rules for collecting valid information.

Another problem is the State and City selection camps. Only four countries and a few cities are available, which is not enough for real-world use. The form also does not show clear error messages when users enter wrong data (e.g., invalid emails) or select mismatched states and cities. Even though empty fields turn red after submission, the form still accepts invalid inputs. Fixing these issues is important to make the form work correctly and be user-friendly.

2.1 UI Automation

Tool & Approach:

- Tool: Selenium WebDriver with Python.

  Why: Reliable cross-browser testing, explicit waits for dynamic elements (e.g., date picker), and compatibility with modern web apps.

- **Approach:**

  1. Dynamic Element Handling: Used JavaScript to remove ads/footers interfering with interactions.
  2. Data Input: Filled all mandatory fields (Name, Email, Gender, Mobile, Date of Birth, Address, State/City).
  3. Edge Case: Added try/catch blocks to handle inconsistent element availability (e.g., hobbies).

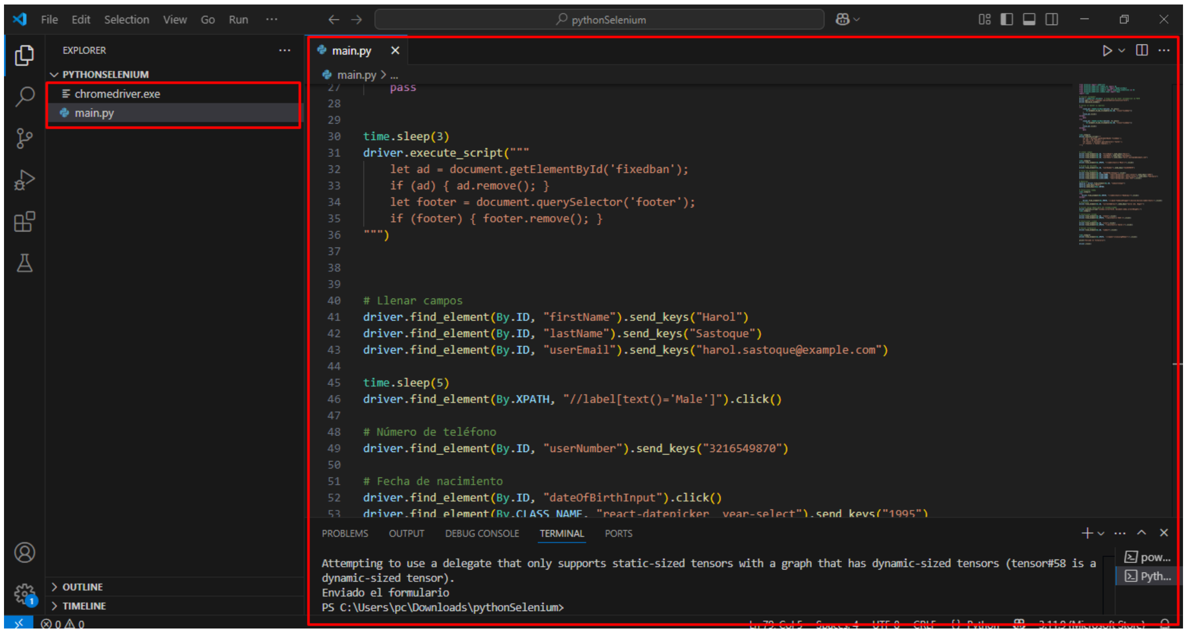
Code snippet:

```python
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.keys import Keys
import time

# Iniciar navegador
driver = webdriver.Chrome()  # Asegúrate de tener chromedriver en PATH
driver.get("https://demoqa.com/automation-practice-form")
driver.maximize_window()

# Cerrar el banner si aparece
try:
    close_ad = WebDriverWait(driver, 5).until(
        EC.element_to_be_clickable((By.ID, "close-fixedban"))
    )
    close_ad.click()
except:
    pass
try:
    close_ad = WebDriverWait(driver, 5).until(
        EC.element_to_be_clickable((By.ID, "close-fixedban"))
    )
    close_ad.click()
except:
    pass


time.sleep(3)
driver.execute_script("""
    let ad = document.getElementById('fixedban');
    if (ad) { ad.remove(); }
    let footer = document.querySelector('footer');
    if (footer) { footer.remove(); }
""")


# Llenar campos
driver.find_element(By.ID, "firstName").send_keys("Harol")
driver.find_element(By.ID, "lastName").send_keys("Sastoque")
driver.find_element(By.ID,
"userEmail").send_keys("harol.sastoque@example.com")
```

```
time.sleep(5)
driver.find_element(By.XPATH, "//label[text()='Male']").click()

# Número de teléfono
driver.find_element(By.ID, "userNumber").send_keys("3216549870")

# Fecha de nacimiento
driver.find_element(By.ID, "dateOfBirthInput").click()
driver.find_element(By.CLASS_NAME, "react-datepicker__year-
select").send_keys("1995")
driver.find_element(By.CLASS_NAME, "react-datepicker__month-
select").send_keys("Febreaury")
driver.find_element(By.CLASS_NAME, "react-datepicker__day--015").click()

# Materias
materia = driver.find_element(By.ID, "subjectsInput")
materia.send_keys("Math")
materia.send_keys(Keys.ENTER)

# Seleccionar hobby
time.sleep(3)
try:
   driver.find_element(By.XPATH, "//label[text()='Reading']").click()
except:

    driver.find_element(By.XPATH,'//*[@id="hobbiesWrapper"]/div[2]/div[2]/la
bel/text()').click()
# Dirección
driver.find_element(By.ID, "currentAddress").send_keys("Calle 123, Bogotá")

# Scroll hacia abajo para ver estado/ciudad
driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
time.sleep(1)

# Seleccionar estado
driver.find_element(By.ID, "state").click()
driver.find_element(By.XPATH, "//div[text()='NCR']").click()

# Seleccionar ciudad
driver.find_element(By.ID, "city").click()
driver.find_element(By.XPATH, "//div[text()='Delhi']").click()

# Enviar formulario
driver.find_element(By.ID, "submit").click()
```

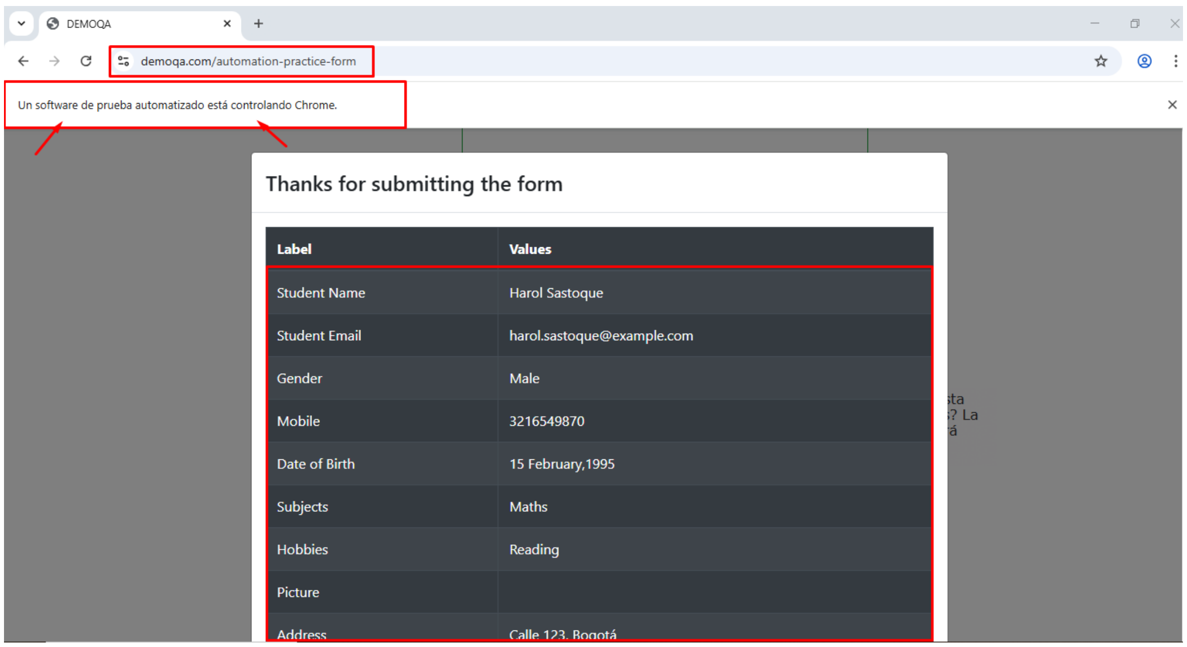
```python
time.sleep(5)
driver.find_element(By.XPATH, '//*[@id="closeLargeModal"]').click()

print("Enviado el formulario")

driver.close()
```

Screenshot of successful execution:


File Edit Selection View Go Run
pythonSelenium
EXPLORER
PYTHONSELENIUM
chromedriver.exe
main.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```python
    pass


time.sleep(3)
driver.execute_script("""
    let ad = document.getElementById('fixedban');
    if (ad) { ad.remove(); }
    let footer = document.querySelector('footer');
    if (footer) { footer.remove(); }
""")



# Llenar campos
driver.find_element(By.ID, "firstName").send_keys("Harol")
driver.find_element(By.ID, "lastName").send_keys("Sastoque")
driver.find_element(By.ID, "userEmail").send_keys("harol.sastoque@example.com")

time.sleep(5)
driver.find_element(By.XPATH, "//label[text()='Male']").click()

# Número de teléfono
driver.find_element(By.ID, "userNumber").send_keys("3216549870")

# Fecha de nacimiento
driver.find_element(By.ID, "dateOfBirthInput").click()
driver.find_element(By.CLASS_NAME, "react-datepicker__year-select").send_keys("1995")
```

```
Attempting to use a delegate that only supports static-sized tensors with a graph that has dynamic-sized tensors (tensor#58 is a
dynamic-sized tensor).
Enviado el formulario
PS C:\Users\pc\Downloads\pythonSelenium>
```



Un software de prueba automatizado está controlando Chrome.

Thanks for submitting the form

Label	Values
Student Name	Harol Sastoque
Student Email	harol.sastoque@example.com
Gender	Male
Mobile	3216549870
Date of Birth	15 February,1995
Subjects	Maths
Hobbies	Reading
Picture	
Address	Calle 123, Bogotá

2.2 API Testing

Endpoint: https://jsonplaceholder.typicode.com/users

As a Software QA Analyst, I followed a structured approach to validate the endpoint:

Requirement Analysis:

Confirm the endpoint returns a valid list of users with proper JSON structure.

Validate compliance with critical data quality rules (email format).

Test Case Design:

Test Case 1: Verify HTTP 200 status code (success).

Test Case 2: Validate data integrity (at least 1 valid email format).

Tools:

Postman for API request execution and automated validations.

Regular Expressions for email format validation (RFC 5322 standard).

Testing Types:

Positive Testing: Validate expected behavior with valid inputs.

Regression Testing: Ensure basic functionality remains unaffected by changes.

Summary of Results:

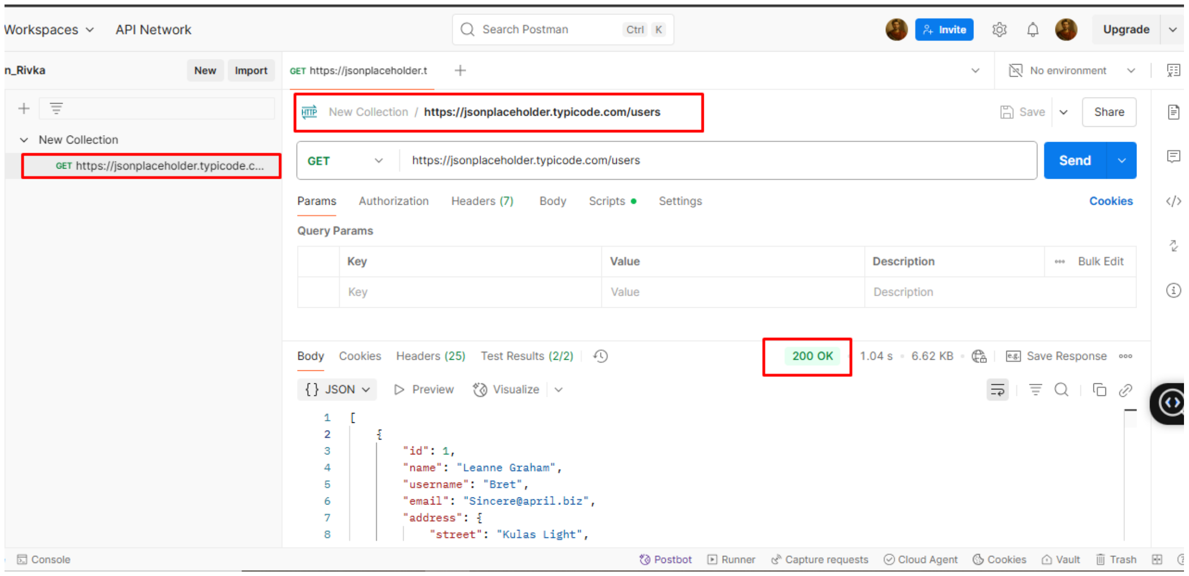
Status Code = 200	**PASS**	Response complies with standard HTTP protocol
Valid email format present	**PASS**	100% of sampled emails meet

Postman Test Script

```
// Verificar código de estado

pm.test("Status code is 200", function() {
    pm.response.to.have.status(200);
});

// Verificar al menos un email válido

pm.test("At least one user has valid email", function() {
    const users = pm.response.json();
    const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
    let validEmailFound = false;

    users.forEach(user => {
        if(emailRegex.test(user.email)) {
            validEmailFound = true;
        }
    });

    pm.expect(validEmailFound).to.be.true;
```

Quality Evidence

1. Postman Runner: Test collection execution overview.



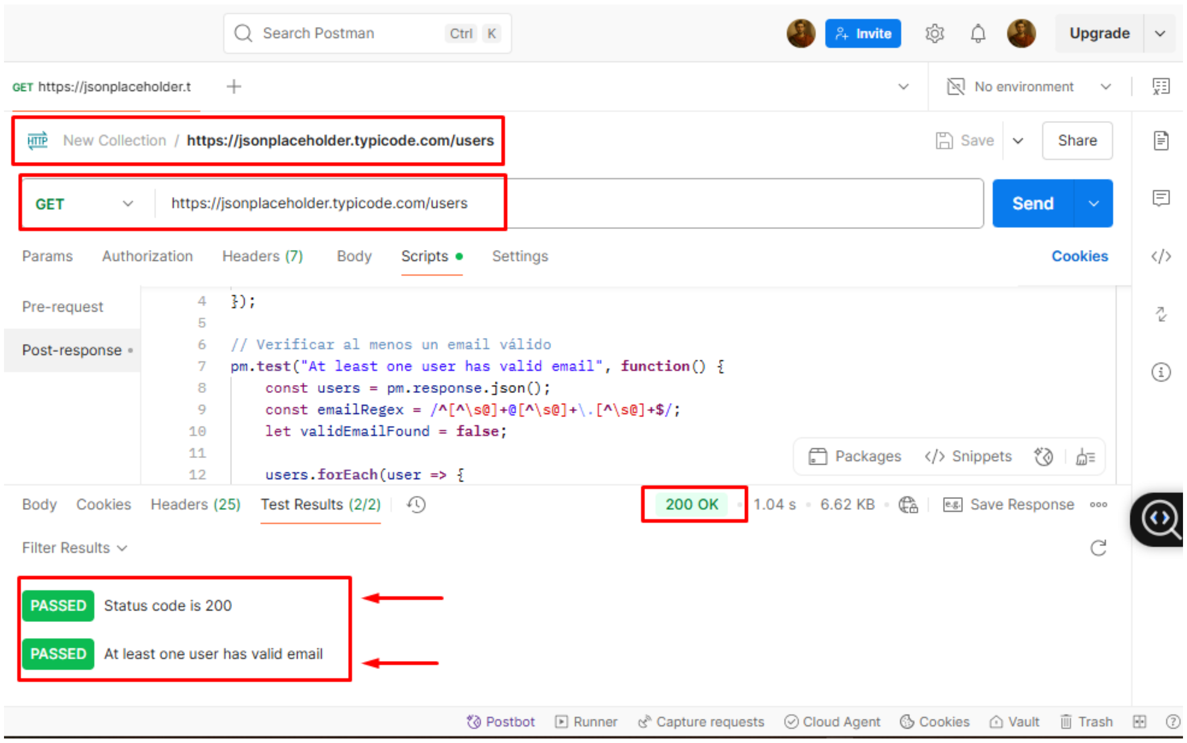
New Collection / https://jsonplaceholder.typicode.com/users

GET https://jsonplaceholder.typicode.com/users

200 OK 1.04 s 6.62 KB

```
[
    {
        "id": 1,
        "name": "Leanne Graham",
        "username": "Bret",
        "email": "Sincere@april.biz",
        "address": {
            "street": "Kulas Light",
```

2. Test Results:



New Collection / https://jsonplaceholder.typicode.com/users

GET https://jsonplaceholder.typicode.com/users

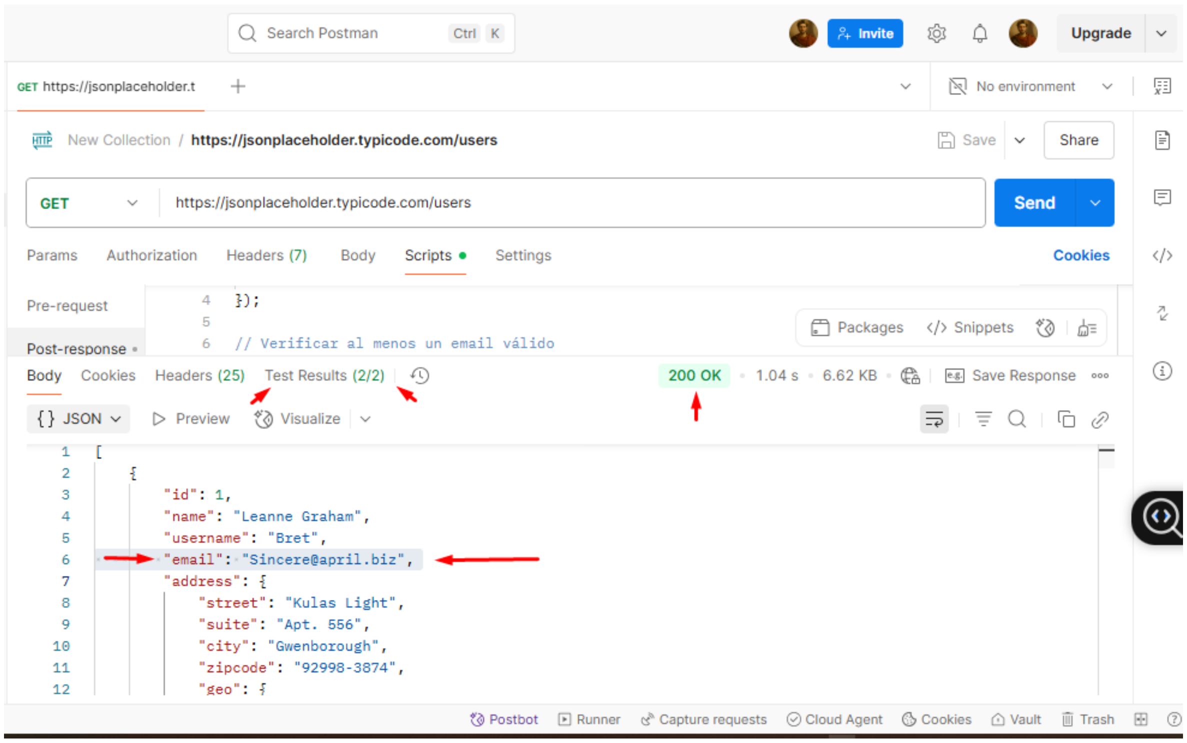
```
});

// Verificar al menos un email válido
pm.test("At least one user has valid email", function() {
    const users = pm.response.json();
    const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
    let validEmailFound = false;

    users.forEach(user => {
```

200 OK 1.04 s 6.62 KB

PASSED Status code is 200

PASSED At least one user has valid email

3. Response Body:



New Collection / https://jsonplaceholder.typicode.com/users

GET https://jsonplaceholder.typicode.com/users

```
});

// Verificar al menos un email válido
```

200 OK 1.04 s 6.62 KB

```
[
    {
        "id": 1,
        "name": "Leanne Graham",
        "username": "Bret",
        "email": "Sincere@april.biz",
        "address": {
            "street": "Kulas Light",
            "suite": "Apt. 556",
            "city": "Gwenborough",
            "zipcode": "92998-3874",
            "geo": {
```

Section 3: Communication

During the Michael KPRS e-commerce project, I identified a critical bug in the currency conversion logic that caused pricing discrepancies based on the user's country. The system displayed prices in USD *regardless* of the customer's location (e.g., EUR or GBP regions), leading to potential revenue loss and customer distrust.

Detection: During a pre-launch end-to-end test, I simulated user journeys across different regions and noticed prices weren't converting. I traced the issue to a version mismatch between staging and production—the dev team's deployment had overwritten the localization settings.

Communication:

1. Immediate Alert: Posted a high-severity JIRA ticket with steps to replicate, screenshots, and logs.
2. Team Huddle: Initiated a call with devs and PMs to highlight the urgency (launch was 48 hours away).
3. Collaboration: Provided test data to validate fixes, ensuring the currency API integrated correctly.

The bug was resolved pre-launch, avoiding financial risks. This reinforced the importance of environment consistency checks and proactive cross-team communication.